AD-A077 896 MCDONNELL DOUGLAS ASTRONAUTICS CO HUNTINGTON BEACH CALIF F/G 9/2
METRICS OF SOFTWARE QUALITY.(U)
AUG 79 Z JELINSKI , P B MORANDA F49620-77-C-0099
UNCLASSIFIED AFOSR-TR-79-0976 NL

1 OF 1
AD
A077896

END
DATE
FILMED
1-80
DDC

MICROCOPY RESOLUTION TEST CHART
NATIONAL BUREAU OF STANDARDS-1963-A

AD A077896

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

(19) REPORT DOCUMENTATION PAGE		READ INSTRUCTIONS BEFORE COMPLETING FORM
1. REPORT NUMBER (18) AFOSR-TR-79-0976	2. GOVT ACCESSION NO.	3. RECIPIENT'S CATALOG NUMBER
4. TITLE (and Subtitle) (6) METRICS OF SOFTWARE QUALITY. AO65196		5. TYPE OF REPORT & PERIOD COVERED Interim
		6. PERFORMING ORG. REPORT NUMBER
7. AUTHOR(s) (10) Z./Jelinski ~~and~~ P.B./Moranda		8. CONTRACT OR GRANT NUMBER(s) (15) F49620-77-C-0099
9. PERFORMING ORGANIZATION NAME AND ADDRESS McDonnell Douglas Astronautics Company 5301 Bolsa Avenue Huntington Beach, CA 92647		10. PROGRAM ELEMENT, PROJECT, TASK AREA & WORK UNIT NUMBERS 61102F 2304/A2 (16) (17)
11. CONTROLLING OFFICE NAME AND ADDRESS Air Force Office of Scientific Research/NM Bolling AFB, Washington, DC 20332		12. REPORT DATE (11) Aug 1979
		13. NUMBER OF PAGES 27 (12) 26
14. MONITORING AGENCY NAME & ADDRESS(if different from Controlling Office) LEVEL II		15. SECURITY CLASS. (of this report) UNCLASSIFIED
		15a. DECLASSIFICATION/DOWNGRADING SCHEDULE

16. DISTRIBUTION STATEMENT (of this Report)

(9) Annual rept., 1 Jun 78 - 31 May 79,

Approved for public release; distribution unlimited.

17. DISTRIBUTION STATEMENT (of the abstract entered in Block 20, if different from Report)

DDC RECEIVED SEP 17 1979 C

18. SUPPLEMENTARY NOTES

19. KEY WORDS (Continue on reverse side if necessary and identify by block number)

Software metrics
Test tools

20. ABSTRACT (Continue on reverse side if necessary and identify by block number)

This report covers the period from 1 June 1978, to 31 May 1979, during which the second of three phases was completed. This phase concentrated on the augmentation of the Program Testing Translator described in the Final Report on AFOSR F44620-74-C-008 of December 1976. The augmentation permits formation of links of greater length than those previously used.

DD FORM 1 JAN 73 1473

UNCLASSIFIED 389310 nt

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

20. Abstract continued.

It also alleviates the process of composing the tracks of execution sequences, a requirement for establishing an automatic record of the status of testing.

Applications have been made to an Air Force logistics model (Optimum Repair Level Analysis-ORLA). In these applications tests were made with the purpose of successively increasing the count of exercised tracks. Because there are numerous user-selected options the count so obtained was not truly indicative of the data base's effect on testing. Nonetheless progress has been made in determining the way to process programs automatically in the way indicated in the Annual Report for the preceding period.

MCDONNELL DOUGLAS ASTRONAUTICS COMPANY

MCDONNELL DOUGLAS

CORPORATION

C

[redacted]
AUGUST 1979

METRICS OF SOFTWARE QUALITY

MCDONNELL DOUGLAS ASTRONAUTICS COMPANY
5301 BOLSA AVENUE
HUNTINGTON BEACH, CALIFORNIA 92647

JULY 1979

ANNUAL REPORT FOR THE PERIOD 1 JUNE 1978 - 31 MAY 1979
CONTRACT F49620-77-0099

PREPARED FOR:

AIR FORCE OFFICE OF SCIENTIFIC RESEARCH
BOLLING AIR FORCE BASE, D.C.
D.C. 20332

ATTENTION: LT. COL. GEORGE W. MCKEMIE, CONTRACT MONITOR
DIRECTORATE OF MATHEMATICAL AND INFORMATION SCIENCES

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

REPORT DOCUMENTATION PAGE		READ INSTRUCTIONS BEFORE COMPLETING FORM
1. REPORT NUMBER AFOSR-TR- 79-0976	2. GOVT ACCESSION NO.	3. RECIPIENT'S CATALOG NUMBER
4. TITLE (and Subtitle) METRICS OF SOFTWARE QUALITY		5. TYPE OF REPORT & PERIOD COVERED 1 June 1978 to 31 May 1979 Interim Report
		6. PERFORMING ORG. REPORT NUMBER
7. AUTHOR(s) Z. Jelinski and P. B. Moranda		8. CONTRACT OR GRANT NUMBER(s) F49620-77-0099
9. PERFORMING ORGANIZATION NAME AND ADDRESS McDonnell Douglas Astronautics Company 5301 Bolsa Ave. Huntington Beach, CA 92647		10. PROGRAM ELEMENT, PROJECT, TASK AREA & WORK UNIT NUMBERS
11. CONTROLLING OFFICE NAME AND ADDRESS Air Force Office of Scientific Research Bolling Air Force Base District of Columbia 20332		12. REPORT DATE July 1979
		13. NUMBER OF PAGES
14. MONITORING AGENCY NAME & ADDRESS(if different from Controlling Office)		15. SECURITY CLASS. (of this report) Unclassified
		15a. DECLASSIFICATION DOWNGRADING SCHEDULE

16. DISTRIBUTION STATEMENT (of this Report)

Approved for public release; distribution unlimited

17. DISTRIBUTION STATEMENT (of the abstract entered in Block 20, if different from Report)

18. SUPPLEMENTARY NOTES

19. KEY WORDS (Continue on reverse side if necessary and identify by block number)

Software metrics
Test tools

20. ABSTRACT (Continue on reverse side if necessary and identify by block number)

This report covers the period from 1 June 1978, to 31 May 1979, during which the second of three phases was completed. This phase concentrated on the augmentation of the Program Testing Translator described in the Final Report on AFOSR F44620-74-C-008 of December 1976. The augmentation permits formation of links of greater length than those previously used.

DD FORM 1 JAN 73 1473 EDITION OF 1 NOV 65 IS OBSOLETE

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

It also alleviates the process of composing the tracks of execution sequences, a requirement for establishing an automatic record of the status of testing.

Applications have been made to an Air Force logistics model (Optimum Repair Level Analysis-ORLA). In these applications tests were made with the purpose of successively increasing the count of exercised tracks. Because there are numerous user-selected options the count so obtained was not truly indicative of the data base's effect on testing. Nonetheless progress has been made in determining the way to process programs automatically in the way indicated in the Annual Report for the preceding period.

SUMMARY

This report covers the period from 1 June 1978, to 31 May 1979, during which the second of three phases was completed. This phase concentrated on the augmentation of the Program Testing Translator described in the Final Report on AFOSR F44620-74-C-008 of December 1976. The augmentation permits formation of links of greater length than those previously used.

It also alleviates the process of composing the tracks of execution sequences, a requirement for establishing an automatic record of the status of testing.

Applications have been made to an Air Force logistics model (Optimum Repair Level Analysis-ORLA). In these applications tests were made with the purpose of successively increasing the count of exercised tracks. Because there are numerous user-selected options the count so obtained was not truly indicative of the data base's effect on testing. Nonetheless progress has been made in determining the way to process programs automatically in the way indicated in the Annual Report for the preceding period.

METRICS OF SOFTWARE QUALITY

Section 1

OBJECTIVES AND TASK DESCRIPTIONS

1.1 WORK ACCOMPLISHED

The primary effort during the contract period can be encompassed by the description of two tasks:

Task I - Experimentation with Software Quality metrics will include:

a. Selection of FORTRAN Programs for strategy testing of techniques identified and investigated in Phase I of this contract.

b. Analysis of the programs using appropriate techniques.

c. Composition of the extended version of the Program Testing Translator, integrating into the pre-processor the means of augmenting program code so as to provide valuations of the program predicates, and values of the artificial program variables which provide the data for the search procedures.

d. Determination of the level of testedness in each of the selected programs, and development of means of automatically selecting test cases to achieve specified paths through the tested programs.

Task II - Development of a methodology for developing candidate drivers for untested regions of a program; this consists of specifying starting points on the input domain which are likely to cause execution sequences which produce designated predicate valuations.

1.2 ADDITIONAL WORK

The goals of the research are to develop a preliminary design and partial implementation of a message entry/computer/display combination for interactive testing and case selection so as to achieve exhaustive testing of arbitrary

(FORTRAN) programs. Studies will also be made to assess the practicality of a fully automated version of testing.

This additional work can be subdivided into the following major tasks:

a. Tailor or expand the testing programs developed and used in Section 2.2.1.1 of the previous work. The necessity and benefits of expanding the procedure to include a larger set of program predicates will be studied.

b. Modify, install and test the tool on a laboratory computer when the scope and size of the general purpose test tool is established. The use of macros, augmenting an intermediate language, will be investigated on a trial basis.

c. Test the tool and the methodology using the several constructs (connection matrix, status vectors, predicate valuations, and input and output data) through the implementation on a "laboratory" type of computer, such as the Nanodata QM-1.

Section 2
STATUS OF RESEARCH EFFORTS

2.1 EXPERIMENTATION WITH SOFTWARE QUALITY METRICS

The selection of useful metrics from the many candidates is discussed in the previous annual report (Reference 1). The application of the metric representing the degree of testedness was initiated in the previous work interval and it was noted in the report that the efficiency of the means then available for establishing the level of testedness, even for simple instruction- or segment-coverage was very poor. The primary effort of the present study is development of automatic aids for determining the coverage provided by the input data sets. The principal product, correspondingly, is the computer program which facilitates the determination of the degree of testedness.

2.1.1 FORTRAN Programs Tested

In order to augment the algorithmic-type programs previously studied, with programs which are more typical of those encountered in the field, a review of some of the Air Force Logistics programs was made. Inspections of several programs were made for the purpose of selecting a useful candidate for coverage testing. A review of the MOD-METRIC model revealed a very complex program which would have provided an excellent candidate because of the diverse modes which can be exercised. However the fact that documentation of the FORTRAN program is almost non-existent in the mid-levels of documentation (between the overview, on the one end, and inserted comments, on the other), the program was passed over. The LSC (Logistics Support Cost) model was not selected because it consists of a set of rather simple algebraic formulas. Another model, LEM (Logistics Effect Model) is not yet widely known in the Air Force, and primarily was eliminated for that reason. The Air Force LCOM (Logistics Composite Model) was investigated and while its basic or underlying language is FORTRAN, it has a language of its own and is not therefore suitable for analysis. Another difficulty with LCOM is that production runs with that model would cost far in excess of any contemplated expenditures for the testing task which was planned. This is so because the model relies on simulations with an underlying SIMSCRIPT II program, to produce Monte Carlo based statistics of operational parameters. The program with greatest potential among those investigated is commonly called ORLA (Optimum Repair Level Analysis). The particular version employed was written by O. R. Johnson of Warner-Robins Air Force Logistics Center.

ORLA employs costs associated with the acquisition,logistic support, and replacement, of airplane subsystems. Three options are generally considered in an ORLA analysis: discard at (suspected) failure of the subsystem; repair of the failed subsystem at the base (home airport), or repair at an Air Force depot (generally supporting several bases). Some 11 different cost components are involved for the latter two options, while 3 cost components comprise the discard option total. Although computations are not complex, and, indeed, the cost components are simple algebraic formulas, the so-called sensitivity analysis presents some interesting complexities and decisions. The aim of this sensitivity analysis is to determine (to the nearest 1% of the baseline value) the point at which the nominal decision, derived from the baseline values, will be reversed. This is accomplished for any of choice from the 17 different input factors, and it provides, as the name indicates, a measure of the sensitivity or stability of the decision in the face of possible changes in or misestimation. The sensitivity analysis is flowcharted in Section 2.1.3 where the application to the entire ORLA program is illustrated.

2.1.2 Augmentation of the Program Testing Translator

The most significant and tangible product of the study is the augmented version of the PTT (Program Testing Translator). This translator is described in MDAC Report M2085074 (Reference 2). PTT was employed in the earlier studies to segment FORTRAN programs, and to obtain basic counts of segment usages for different input data sets. Segments were initially formed at explicit fork-or branch-points such as IF's or multiple GOTO's. In the augmentation one of the salient features is the inclusion of the implicit predicates, such as DO loop exits.

This one improvement provided by the augmented PTT, APTT, makes it more nearly possible to trace program flows. However, the concept which is employed and which is based on so-called program tracks, does not require that such precise traces be made. It is, accordingly, the fact that a predicate representing a branch point is created which is of importance.

In addition to automatically segmenting FORTRAN programs and monitoring the usages by the so formed segments, APTT ferrets out unreachable/unexercisable segments before the program is run, that is to say, from static analysis of the code.

Additional work was required to tailor the original translator to more efficient usage, some vestiges of the PET program, from which PTT and APPT have been derived, were eliminated.

Algorithms for developing the concatenation of instructions in the manner described in Reference 1 were developed. These linkings emphasize the dynamic nature of the program, and list the sequence of steps which the computer would process; for example, the instruction associated with a label of a GOTO is assumed to be exercised (absolutely, unless it is a branching type of instruction).

The format of the primary APTT output report facilitates the identification of unexercised branches after a set of runs have been made.

2.1.3 Analysis of Programs

The main ORLA program consists of 488 lines of FORTRAN code (each branch of branching instructions are counted). Briefly the components of ORLA can be described by the following: Initialization (about 15 instructions); Read Constants (64); Compute Failure Rate (59); Computation of Aerospace Ground Equipment Usage (66); ORLA Variable Identification (34); Economic Analysis (33); Write Summary (15); Computation Routine (58); Rank Economic Values (22); Sensitivity Analysis (93); Write Repair Summary (12). (In addition three peripheral and non-essential subroutines are included in the program: two are merely messages for the user in case he requires explanations of the program, the third is set of error messages in case of inconsistencies in the data. These subroutines are not included in the discussion which follows.)

To drive the basic ORLA a total of 54 variables are employed. These variables provide descriptions of all the logistics involved in acquiring, shipping, repairing, maintaining, and resupplying an aircraft subsystem. Included are variables which represent overhead, such as, training of maintenance personnel, management of inventory, and facilities. The 54 variables are divided into 2 main classes. First a set of 36 variables describe the rates which hold or are projected to hold for the time of the analysis, the force size and deployment scheme, labor and material rates, and so forth. In addition to these, a second class bears directly on the item or subsystem analyzed (ORLA'd); there are 17 variables in the class and they describe, cost and weight of the subsystem and its parts, repair time, and the documentation, training, and special facilities which are required for the item.

In addition to these basic variables there are 10 additional variables which are derived from intermediate computations which rely either on keyboard entry (of parameters relating to the MTBF) or on sharing of resources by several items (AGE or test equipment which is employed or several different subsystems of the aircraft for example). The reason for identifying them with the input variables is that they also can be subjected to the sensitivity analysis.

As noted earlier the ORLA program employs the input values associated with a given item and computes the costs which would be incurred under the three options(discard, repair at base,repair at depot). On the basis of the three ranked costs the optimum or least cost repair level can be determined. Although the numerical values of the costs of the various components of cost are printed out and an indication of the assurance or firmness of the decision which the program makes can be made from these magnitudes, a better measure of the firmness of the decision can be made by use of sensitivity analyses. Each run a set of up to 10 user-selected variables can be identified for use in this analysis. As noted before, the primary purpose is to determine, from variations in the costs due to changes in the selected variable, the point (a percentage of nominal value) where the decision based on nominal or baseline costs is reversed. This is determined to the nearest percentage on the range 20% to 500% (1/5 to 5 times nominal). Should no change in decision occur over this range, the decision is clearly stable with respect to the variable inspected.

Certain variables are known to affect certain options more than others and a user wishing to test for coverage could be guided by this a priori knowledge. Some of this kind of knowledge is also used in the construction of cases which are discussed here. This is counter to the mode which would be used in the final testing scheme where it is assumed the user is unaware of the relationship between input and any particular program segment. In the final version each variable would be varied at random to provide an initial coverage; subsequent coverage would be initiated by a specification of a program path or track, then continued by invoking a search procedure on the input data, and hopefully consumated by an identification of a point which produces an execution which includes selected path or track. Because the status of the study has not progressed to the point where automatic insertion and data generation are possible the procedure used in the example relies on knowledge of the program.

It is cumbersome to illustrate the usage of APTT on the entire ORLA program, but a good indication of the way APTT can be applied in static analysis can be provided by use and inspection of a compact portion of the listing. In Figure 1 is a flow chart of the portion of the program called Reversal Analysis. This is used in part of the sensitivity analysis to compare and rank the costs of the three options.

Application of APTT in static composition of segments from the coding of the above identified program portion is effected by first numbering the instructions as shown in Figure 2. This shows the numbering in the leftmost column and these are associated with the instruction on the right. Labels shown correspond to the original listing and are employed in the flowchart of Figure 1. Thus 396 corresponds to the labelled (215) instruction, JSEN(1)=KDT, at the top of Figure 1, 415 corresponds to the predicate, $\boxed{\text{NUMK(1)-NUMJ(1)=0}}$, which appears just after the labelled 310 CONTINUE instruction in the figure. The APTT segmentation of the program in the above described region is shown in Figure 3.

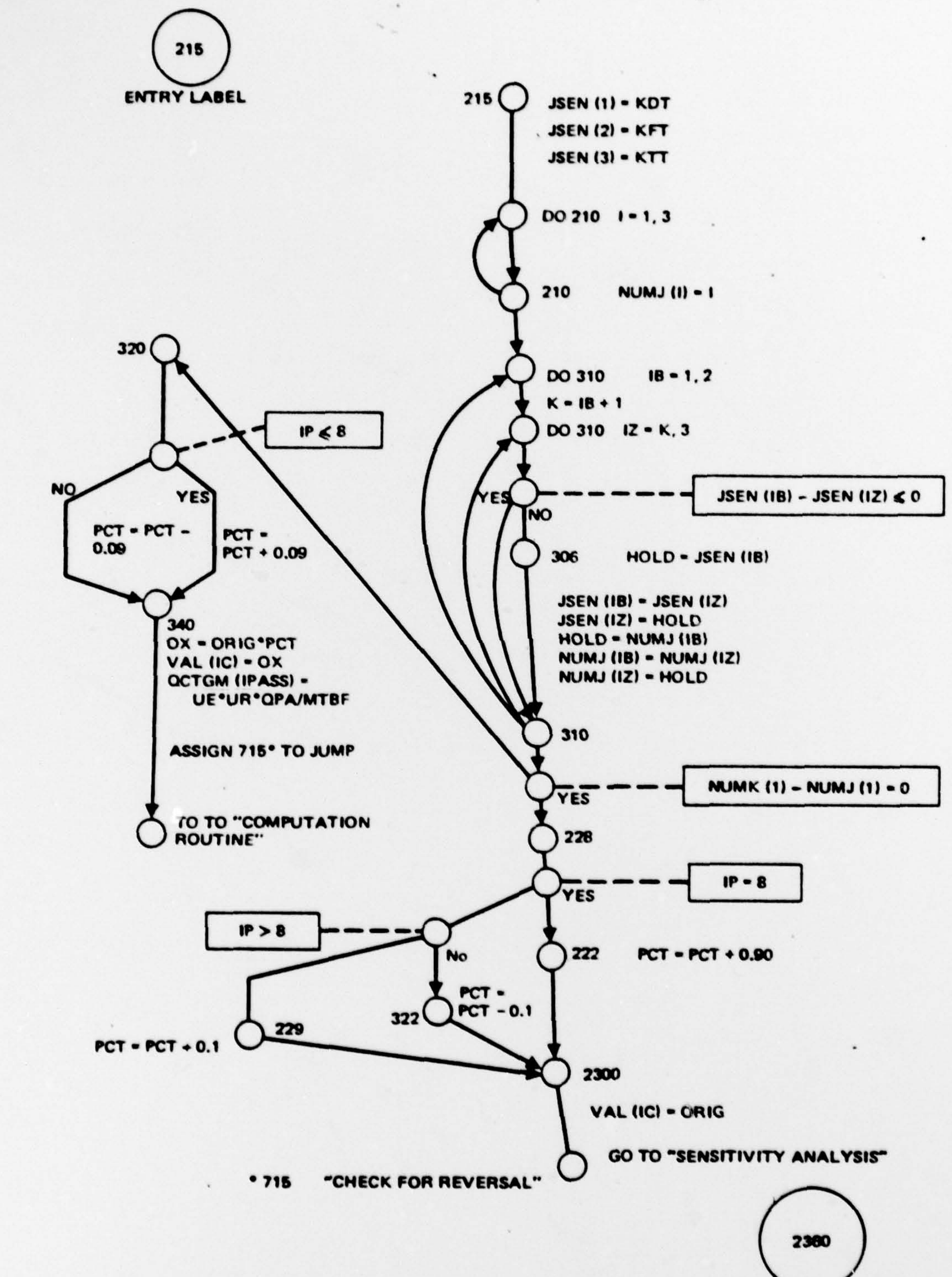

Figure 1. Reversal Analysis

REVERSAL ANALYSIS

```
396            215 JSEN(1) = KDT
397                JSEN(2) = KFT
398                JSEN(3) = KTT
399                DO 210 I = 1,3
400-401        210 NUMJ(I) = I
402            300 DO 310 IB = 1,2
403                K = IB +1
404                DO 310 IZ = K.3
405            305 IF (JSEN(IB)-JSEN(IZ)) 310,310,306
406            306 HOLD=JSEN(IB)
407                JSEN(IB) = JSEN(IZ)
408                JSEN(IZ) = HOLD
409                HOLD = NUMJ(IB)
410                NUMJ(IB)=NUMJ(IZ)
411                NUMJ(IZ)=HOLD
412-414        310 CONTINUE
415                IF (NUMK(1) = NUMJ(1)) 320,228,320
416            228 CONTINUE
417                IF (IP-8) 322,222,229
418            222 PCT = PCT-.90
419                GO TO 2300
420            322 PCT = PCT-.1
421                GO TO 2300
422            229 PCT = PCT + .1
423                GO TO 2300
424            320 CONTINUE
425                IF (IP-8) 375,375, 360
426            375 PCT =PCT + .09
427                GO TO 340
428            360 PCT = PCT -.09
429            340 OX = ORIG * PCT
430                VAL(IC) = OX
431                QCTGM(IPASS)=VAL(31) * VAL(32) * VAL(46)/VAL(56)
432                ASSIGN 715 TO JUMP
433                GO TO 100
```

Figure 2. APTT Numbering for Program

MDAC SEGMENT XLATOR

T_1	[396-401)
T_2	[401,399-401)
T_3	[401-405)
T_4	[405,412-413)
T_5	[413,404-405)
T_6	[413-414)
T_7	[414,402-405)
T_8	[414-415)
T_9	[415,424-425)
T_{10}	[425-427,429-433,304-319)
T_{11}	[425,428-433,304-319)
T_{12}	[415-417)
T_{13}	[417,420-421,460-461)
T_{14}	[417-419,460-461)
T_{15}	[417,4220-429,460-461)
T_{16}	[405-413)

Figure 3. APTT Segment Identification

It is important to note that in most cases the segments are made up of several of the earlier-defined PTT segments. Those segments were truncated by labels, GOTO's,etc. Several other points require explanation. First the segments identified with the bracket/parentheses, start with an instruction number which is either the start of the program, or subroutine, or a predicate (IF statement in most cases), the remainder of the numbers in the sequence denote the instructions which will be executed in sequence, the end of the sequence of numbers is identified by a number corresponding to a predicate or branch point. Thus T_1 starts with the labelled instruction 396, then in turn by 397, 398, 399, 400 and ends with 401. The instruction 401 is an implied predicate $\boxed{\text{DO LOOP END=TRUE}}$. If the predicate is true the next segment taken is T_3 which describes passage from the DO210 loop to the DO310 loop, and continuation to the next predicate which is an explicit predicate, $\boxed{\text{JSEN(IB-JSEN(IZ)=0}}$. If the predicate is false, the segment T_2 is executed, with an initial 401, then entry or reentry into the DO210 loop at 399.

Because of the selection of only a portion of the program, some segments shown in Figure 3, such as T_{10}, list instructions which are outside the range of those shown in Figure 2. The explanation for T_{10} which will be given here should serve for others as well. T_{10} is made up of [425-427, 429-433, 304-319), and it is the last group that is out of range. Instruction 433 is a GOTO 100 sintruction and the APTT number 304 corresponds to the label 100, which is the start of the so-called Computation Routine. This routine computes cost components for the three options and the 304-319 segment is the initial segment of that routine. A "return" to the portion which is displayed in Figure 1 is at the end of the Computation Routine (at APTT number 355-not shown). There is one entry point from the Computation Routine and that is at T_1. So far as the local analysis is concerned T_{10} joins to T_1. The original set of segments can be tailored so as to exhibit only local connections by the above method. So far as the illustration of technique is concerned, however, there is no need to work at that level of detail.

Figure 1 shows the two major exits: computation routine (label 100 and APTT number 304); sensitivity analysis (label 2360, APTT Number 384);

The program was initially driven with a set of standard elements for one of the Air Force's aircraft types and an imaginary subsystem. The program's pre-selected variables were used in the sensitivity analysis (these correspond to the size of the force being fitted, the number of hours per month the aircraft will be used, the repair manhours, the unit cost, the Mean Time Between Failures, the cost of depot AGE, and the cost of base level AGE). The initial data exercised the segments listed in Figure 3 as follows.

Segment	Number of Executions
T_1	118
T_2	236
T_3	118
T_4	241
T_5	118
T_6	236
T_7	118
T_8	118
T_9	5
T_{10}	3
T_{11}	2
T_{12}	113
T_{13}	28
T_{14}	4
T_{15}	81
T_{16}	113

For this arbitrary set of data all explicit and implicit predicates were exercised. This complete (local) testing was fortuitous in a sense, for in three successive runs with other data T_{10} was not exercised, while T_9 and T_{11} were not exercised in one case.

The static aspects of APTT are well illustrated by the foregoing. The dynamic aspects can be illustrated by the results from four data sets. The first or nominal is the set identified above, the second maintained the same standard elements and changed one item parameter, the unit cost (from 3600 to 36). The third restored the unit cost to the original value and changed one standard element, depot labor rate (from 12.44 to 1). The fourth changed the turnover rate from .15 to 15.

Results over the entire 113 segments of the program show that the initial choice of data was indeed exceptional, since 96.63% (86 out of 89) of the segments exercised by the four segments were exercised by the initial set.

The change in cost by a factor of 100 (the second case) exercised two segments not exercised by the first set and these correspond to predicate branches caused by the re-ranking of the costs of the three options (discard would be the least expensive). Similarly for the fourth set, a reversal of the costs of depot and intermediate repair is effected by the extreme value chosen for depot turnover.

Examination of the 113 segments comprising the ORLA program, shows that 24 segments were unexercised by the four simple cases. But, of these, 13 depend on choices which are prompted by program; that is they are yes/no responses to questions concerning choices as to whether the user wishes to correct an entry, whether he wishes an explanation, whether he wants to run a batch of several items, etc. In some cases these choices reflect into the substance of the program and in others they stimulate isolated calls and returns without exercising any computations. Of the 11 segments which remain, all but four can be exercised with data. These four are associated with CONTINUE's at the ends of DO loops and they do not appear to be reachable (but they must be present in order for a compilation to take place).

As a very simple and brief explanation of the actual technique used for constructed cases, and as a useful means of discussion of the automated version of the process, the predicate, $\boxed{EOQ<A}$, involving the two program variables EOQ and A will be discussed.* The APTT post-processor tally usages of the entire program shows that the true branch of this predicate is taken on every encounter (1143 passages in the 4 cases). The code contiguous to the predicate shows that the true branch corresponds to the inequality:

*These variables occur in the Computation Subroutine and represent Economic Order Quantity (EOQ) and a "Pipeline" content (A).

$$4.4\sqrt{A} < A$$

or

$$19.36 < A.$$

Again by use of other parts of the code, it is established that

$$A = 12.V_{45}.V_{48}.V_{31}.V_{32}.V_{46}/V_{56}$$

where the V's are all input variables.
Thus the choice $V_{45} = 0$, among many others, will cause the false branch to be taken.

It is well to note that in the contemplated scheme, random numbers would be used over convenient ranges for all of the input variables, and, in this case, the probability of producing an A value less than 19.36 would be extremely high. Thus it is highly likely that the case investigated here would not have arisen in the context of an unexercised branchat a corresponding stage of testing.

Should a similar predicate branch be untested after an initial set of data runs, the following procedure, discussed at length in Reference 1 , would apply. The augmenting program variable C=EOQ-A would be inserted at the predicate site during the APTT pre-processing. During each pass the value of C would be evaluated (in combination with other inserted augmenting variables at other sites of predicates). Variations on the input data would be made according to a search scheme until a point is reached where all augmenting variables have the desired sign - in the present case, C must be positive.

2.1.4 Determination of Level of Testedness

Only segment level testing has been gauged by the testedness criterion. The technique of generating program tracks automatically, which has yet to be implemented, will permit testedness measures of the path-or track-coverage.

2.2 DEVELOPMENT OF CANDIDATE DRIVERS

This task has been postponed until the APTT has embodied the features described above. Expenditures on the program have been nearly as planned for Task I and Task II can be carried out in the next year under the present funding.

Section 3
PUBLICATIONS

3.1 STATUS OF PREVIOUSLY REPORTED PUBLICATIONS

1. P. B. Moranda, "Event-Altered Rate Models for General Reliability Analysis", this was re-routed from a special issue on Software Reliability to the regular issue of IEEE transactions on Reliability (December 1979).

2. P. B. Moranda, "Probability-Based Models for the Failures During Burn-in Phase" still under review for publication in the Journal of the Operations Research Society of America.

3. P. B. Moranda, "Asympototic Limits to Program Testing", INFOTECH State of the-Art Report on Program Testing, INFOTECH 1979.

3.2 NEW WORK IN PROGRESS OR PUBLISHED

1. P. B. Moranda, "Comments On: Competing Software Reliability Models", to appear in IEEE Transactions on Software Reliability (36 pages in draft form).

2. P. B. Moranda, "Comments on: How to Measure Software Reliability and How Not To", to be submitted to IEEE Transactions on Reliability.

3.3 COMMENTS AND REVIEWS

1. P. B. Moranda, comments on article by A. Fitzsimmons and T. T. Love in Surveyors Forum, ACM Computing Surveys, December 1978.

2. P. B. Moranda, Review of "A Controlled Experiment in Program Testing and Code Walkthroughs/Inspections", by G. J. Myers, Computing Reviews, February 1979.

Section 4

ZYGMUNT JELINSKI - Branch Chief, Computer Science

University of London:	B.S., Economics and Statistics (1952)
Metropolitan College, England:	M.B.A., Business Administration (1953)
University of London:	M.S., Mathematical Statistics (1954)

Mr. Jelinski has been managing computer research and development programs for 23 years. As Branch Chief, Computer Sciences, he currently directs research in software reliability, modeling, validation and verification, language processing, emulation and simulation. Software tools developed and/or maintained under his direction include the Program Evaluator and Tester (PET), the SUMC Meta-assembler, the Compiler Writing System, the OPAL Development Tool (OPALDET), CAMIL, and the Generalized Language Processor (METRAN). He was Study Manager, or a contract to develop methodology for effective test case selection (a research contract for the National Bureau of Standards) and he directed a study for NASA on methodology for reliable software. Another program under his direction was the Company-sponsored malfunctions were determined and software validation methodology was develpped and applied to a tactical software system, resulting in accurate prediction of software malfunctions. He also directed the Software Reliability Study sponsored by the Air Force Office of Scientific Research. This study involved research into the development and evaluation of mathematical models representing the pattern of software malfunctions. At Rockwell International Mr. Jelinski was Chief of Systems Programming Technology. In this capacity, he directed the design and retrieval systems, and engineering design aids. Earlier, he managed all programming for the RECOMP II and III computers. At Philco Corporation, he was manager of Programming in support of Philco 2000 computer marketing, and at RCA he was Manager of Applications for RCA 4310 communication computers.

Mr. Jelinski's publications include the following:

HOLDET - Higher Order Language Development and Evaluation Tool (coauthor), MDAC Paper WD 2769, presented to Computers in Aerospace Conference, Los Angeles, November 1977 (AIAA,NASA,IEEE,ACM).

An Approach to Solution of Problems with Support Software as Deliverables, MDAC Paper WD 2759, presented to Defense Systems Management Review, Ft. Belvoir, Virginia, March 1978.

Recent Software Development Techniques in the United States, MDAC Paper WD 2706, presented to Polish Academy of Sciences, Warsaw, September 1976.

Software Reliability Predictions, with Dr. P. B. Moranda, MDAC Paper WD 2482, presented to the Federation for Automatic Control, Boston, and published in its proceedings, August 1975.

Can Statistics be Applied to Software - Historical Perspective, MDAC Paper WD 2531,presented to the Computer Science and Statistics 8th Annual Symposium on the Interface, Los Angeles, February 1975.

Applications of a Probability-Based Model to a Code-Reading Experiment, with Dr. P. B. Moranda, MDAC Paper WD 2067, presented to the Symposium on Software Reliability Sponsored by IEEE, New York, and published in its proceedings, April-May 1973.

Generalized Events-Oriented Simulation System (GESS) - A Performance Evaluation Tool, with Dr. G. S. Chung, MDAC Paper WD 2033, presented to Computer Performance Evaluation Users Group sponsored by National Bureau of Standards, Washington, D.C., published in proceedings, October 1972.

Software Reliability Research, with Dr. P. B. Moranda, MDAC Paper WD 1808, presented to the Conference on Statistical Methods for Evaluation of Computer Systems Performance, Providence, R.I., and published in its proceedings, November 1971.

PAUL B. MORANDA - TECHNICAL ADVISOR
AB, Chemistry, 1942 , Fresno State College
MA, Mathematics, 1948, Ohio State University
PhD, Mathematics, 1953, Ohio State University

McDonnell Douglas Position: Information Systems Advisor, Senior
Dr. Moranda was the Principal Investigator of a contract with AFOSR to investigate the development of Quantitative Methods for Software Reliability. He is also working on Software Validation. During the first nine months of his employment at MDAC he was Principal Investigator for the Software Reliability IRAD. During this period he developed mathematical models for software discrepancies and applied it to failure data to obtain estimates of the error content of software packages, and estimates of their period of error-free performance. For three years subsequent to that assignment, he analyzed and developed logistics model for the YC-15 STOL aircraft.

Previous Experience: Prior to joining MDAC in 1971, Dr. Moranda was Manager of Systems Analysis at Computer Real Time Systems of Newport Beach. Prior to this, he was employed by North American Rockwell for a period of six years. Before joining Computer Real Time Systems, he was Technical Advisor to the Director of Data Management Systems at Autonetics' Information Systems Division. During this interval he participated in several systems studies and in-house development efforts. In the field of transportation, he was responsible for developing a system framework for the analysis of advanced marine transportation systems. Additionally, he participated in: a quantitative analysis of the operations of the over-the-counter trading department of Merrill, Lynch, Pierce, Fenner and Smith; an overview study of the American Stock Exchange; and a systems analysis of the U.S. Federal Court System.

In 1967 he was appointed Scientific Advisor to the Director of Management Systems in which capacity he developed methods of economic forecasting of sales and other business parameters employing random wavelet concepts. This work led to the formulation of a simulation model which predicted, in a balanced way, the sales, profit, cash flow, headcount, backlog, and facilities requirements for the entire corporation.

He devoted full time to work on the California Integrated Transportation Study, performing system synthesis, trade-off studies, and mathematical analyses. In recent years, he has presented five lectures in the transportation field at leading universities: in October of 1966, he presented a lecture on "Advanced Concepts in Transportation Planning" at Carnegie Institute of Technology; in June of 1966, and again in June of 1967, he lectured on the "Application of Systems Analysis to Large Scale Systems - Transportation" at the University of California at Los Angeles; in the Spring quarter of 1967 he organized and administered a full time upper division course in Dynamic Modeling for the University of California at Berkeley, in which he also delivered two lectures, including a summary of the California Transportation Study and the application of analytical techniques to the study of transportation problems.

Prior to joining Autonetics, Dr. Moranda was at the Aeronutronics Division of the Philco Ford Company, engaged in operations analysis of various projects. On a special assignment to the Ford Motor Company, he was responsible for mathematical modeling of the complete automobile production process. Other assignments included development of a methodology for handling fragmentary and unreliable data in a damage assessment center and development of a war game model for assessing military missile effectiveness. In a separate assignment, he held the position of Manager of Systems Analysis for three years.

Section 5
INTERACTIONS

5.1

P. B. Moranda attended the 1st Minnowbrook Workshop on Software Performance Evaluation, sponsored by Syracuse University and Rome Air Development Center. Participated by presenting a paper on modelling and by taking part in two panel discussions on software modeling and metrics.

5.2

Z. Jelinski attended the NSIA Software Conference at Buena Park, California on February 13-15, 1979. He was a panelist on Error/Failure Data Collection Panel chaired by Jack Cooper and he was also a panelist on Firmware Design and Control Issues Panel chaired by Fletcher Buckley.

5.3

Z. Jelinski is an active participant in the Computer Science Software Task Group of the EIA G33 Committee and attends four meetings a year. His specific contributions are in the area of Software Quality and Software Management Techniques.

Section 6
REFERENCES

1. P. B. Moranda. Quantitative Methods for Software Reliability Measurements. McDonnell Douglas Astronautics Company, MDC G6553, Final Report on AFOSR F44620-74-C-008, December 1976.
2. L. Stucki, Program Evaluator and Tester: PET. MDAC M2085074, 1974.
3. Z. Jelinski. Configuration Management and Software Development Techniques. Defense Systems Review 1979 Summer Issue. Fort Belvoir, VA.